XXIV SIMPEP
SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO
08, 09 e 10 Novembro 2017
CONTRIBUIÇÕES DA ENGENHARIA DE PRODUÇÃO PARA UMA ECONOMIA DE BAIXO CARBONO

AVALIAÇÃO DO DESEMPENHO COMPUTACIONAL DE UM PROCEDIMENTO HEURÍSTICO PARA A RESOLUÇÃO DO PROBLEMA DE CORTE DE ESTOQUE

DOUGLAS NOGUEIRA DO NASCIMENTO - douglasnn@fc.unesp.br
UNIVERSIDADE ESTADUAL PAULISTA - UNESP - BAURU-FEB

ADRIANA CRISTINA CHERRI - adriana@fc.unesp.br
UNIVERSIDADE ESTADUAL PAULISTA - UNESP - BAURU-FC

EDILAINE MARTINS SOLER - edilaine@fc.unesp.br
UNIVERSIDADE ESTADUAL PAULISTA - UNESP - BAURU

Área: *6 - PESQUISA OPERACIONAL*
Sub-Área: *6.1 - PROGRAMAÇÃO MATEMÁTICA*

Resumo: *OS PROBLEMAS DE CORTE DE ESTOQUE CONSISTEM EM CORTAR UM CONJUNTO DE OBJETOS DISPONÍVEIS EM ESTOQUE EM UM CONJUNTO DE ITENS PARA ATENDER A DEMANDA DE CLIENTES OU PARA COMPOR ESTOQUE. NESTE TRABALHO, PRETENDE-SE AVALIAR O DESEMPENHO COMPUTACIIONAL DA IMPLEMENTAÇÃO DE UM PROCEDIMENTO HEURÍSTICO UTILIZADO PARA RESOLVER ESSE PROBLEMA, UTILIZANDO DIFERENTES CONFIGURAÇÕES DO SOLVER CPLEX. TESTES COMPUTACIONAIS FORAM REALIZADOS COM INSTÂNCIAS GERADAS ALEATORIAMENTE E MOSTRAM UMA VARIAÇÃO CONSIDERÁVEL NO TEMPO COMPUTACIONAL DE RESOLUÇÃO, NA QUALIDADE DAS SOLUÇÕES E NO NÚMERO DE ITERAÇÕES REALIZADAS EM DIFERENTES COMBINAÇÕES DE CENÁRIOS.*

Palavras-chaves: *PROBLEMA DE CORTE DE ESTOQUE; PROCEDIMENTO HEURÍSTICO; DESEMPENHO COMPUTACIONAL.*

EVALUATION OF THE COMPUTATIONAL PERFORMANCE OF A HEURISTIC PROCEDURE TO SOLVE THE CUTTING STOCK PROBLEM

Abstract: *CUTTING STOCK PROBLEMS CONSIST OF CUTTING A SET OF OBJECTS AVAILABLE IN STOCK TO PRODUCE A SET OF SMALLER ITEMS IN ORDER TO MEET CUSTOMER DEMAND OR TO FILL STOCK. IN THIS PAPER, WE INTEND TO EVALUATE THE COMPUTATIONAL PERFORMANCE OF A HEURIISTIC PROCEDURE USED TO SOLVE THIS PROBLEM, BY USING DIFFERENT CONFIGURATIONS OF THE SOLVER CPLEX. COMPUTATIONAL TESTS WERE PERFORMED WITH INSTANCES RANDOMLY GENERATED AND SHOWED A CONSIDERABLE VARIATION IN THE EXECUTION TIME, QUALITY OF THE SOLUTIONS AND NUMBER OF ITERATIONS PERFORMED IN DIFFERENT COMBINATIONS OF SCENARIOS.*

Keyword: *CUTTING STOCK PROBLEMS; HEURISTIC PROCEDURE; COMPUTATIONAL PERFORMANCE.*

1. Introdução

O problema de corte de estoque (PCE) consiste em cortar um conjunto de objetos padronizados (comprados de fornecedores) disponíveis em estoque em um conjunto de itens com a finalidade de atender demandas de clientes. As quantidades e tamanhos dos itens são especificados e, para solucionar o problema, deve-se otimizar uma função objetivo como, por exemplo, minimizar o número total de objetos cortados, minimizar o custo de cortar objetos, maximizar o lucro ou minimizar a perda de material. Esse tipo de problema ocorre em diversos processos industriais, como o corte de bobinas de aço e de papel, peças de couro, chapas de vidro, peças de madeira, barras de ferro, entre outros tipos de materiais. A importância econômica e a dificuldade de resolução são motivadores para a busca por boas soluções para estes problemas.

Uma solução para o PCE, também chamada de plano de corte, é composta por um conjunto de padrões de corte, que definem quantos e quais itens serão cortados a partir de um objeto, e suas respectivas frequências (quantos objetos devem ser cortados segundo cada padrão de corte), para produzir determinado subconjunto de itens.

Uma das principais dificuldades na resolução do PCE é determinar, de forma ótima, a frequência com que cada padrão de corte deve ser utilizado. A frequência de um padrão de corte deve ser um número inteiro e, dependendo do número de variáveis do problema, pode ser inviável computacionalmente resolver o PCE por métodos de solução exatos. Nestes casos, recorre-se a procedimentos heurísticos que, embora não forneçam uma solução ótima para o problema, obtêm soluções próximas da otimalidade.

Na década de 1960, surgiram as principais pesquisas sobre os PCE. Em Gilmore e Gomory (1961), os autores resolveram, pela primeira vez, um problema real de corte unidimensional, aplicando um procedimento que utiliza o método simplex com geração de colunas em um modelo de otimização linear. Em Gilmore e Gomory (1963), foi apresentado um novo método para a resolução do problema da mochila, que é um procedimento utilizado para gerar as novas colunas (padrões de corte) para o PCE.

Haessler (1975) apresentou uma heurística para resolver problemas de corte de estoque unidimensionais, considerando custos fixos associados a cada troca de padrões de corte. Em Haessler (1980), foram propostas alterações nos procedimentos de Gilmore e Gomory (1961, 1963), as quais indicaram que, controlar a geração dos padrões com limitações do número de itens, ajuda a reduzir problemas de arredondamento e mudanças de padrões de corte. Hinxman (1980) fez uma revisão dos problemas e métodos de resolução dos PCE e

formalizou a heurística de repetição exaustiva, muito utilizada na prática, principalmente quando a demanda dos itens é baixa.

Na década de 1990, Stadtler (1990) propôs um novo método baseado no processo de geração de colunas (Gilmore e Gomory, 1961, 1963), apresentando um procedimento de arredondamento para obtenção de uma solução inteira (heurística residual). Wascher e Gau (1996) realizaram um estudo computacional, reunindo vários procedimentos heurísticos para a resolução do PCE. Vahrenkamp (1996) fez um estudo comparativo entre o método de Gilmore e Gomory (1961) para a solução do PCE e uma heurística baseada em algoritmos genéticos, relacionando a característica randômica dos algoritmos genéticos a uma escolha randômica de padrões de corte, visando minimizar a perda de material.

Cherri et al. (2009) realizaram alterações em heurísticas clássicas da literatura para resolver uma variação do PCE, que considera o aproveitamento de sobras, ou seja, a possibilidade de que retalhos (sobras de objetos cortados) retornem ao estoque para serem utilizados no futuro. Cherri et al. (2013) modificaram as heurísticas propostas em Cherri et al. (2009) e, além de minimizar a perda, assume-se que retalhos em estoque devem ter prioridade de uso durante o processo de corte.

Neste trabalho, um procedimento heurístico para resolver o PCE unidimensional é utilizado. Toda a implementação foi feita utilizando a linguagem OPL (Optimization Programming Language) do software CPLEX. Essa ferramenta utiliza o método *branch-and-cut* para resolver modelos de programação inteira mista, ou *mixed integer programming* (MIP). Configurações foram alteradas para verificar o desempenho computacional do CPLEX em diferentes cenários.

Este trabalho está organizado como segue: na Seção 2 o PCE é apresentado juntamente com o modelo matemático utilizado. Na Seção 3 é descrito o procedimento heurístico para obtenção de soluções inteiras. Os testes computacionais e seus resultados são apresentados na Seção 4. E a Seção 5 destina-se às conclusões deste trabalho.

2. Problema de Corte de Estoque Unidimensional

No problema de corte de estoque (PCE) unidimensional considerado neste trabalho um conjunto de itens demandados deve ser cortado a partir de objetos maiores com tamanhos padronizados (objetos comprados de fornecedores) de modo a minimizar o desperdício.

A Figura 1 ilustra um PCE unidimensional em que são demandados 4 tipos de itens, nas quantidades $d_1 = 2$, $d_2 = 6$, $d_3 = 2$ e $d_4 = 5$. Os itens podem ser cortados a partir de

diferentes tipos de objetos.

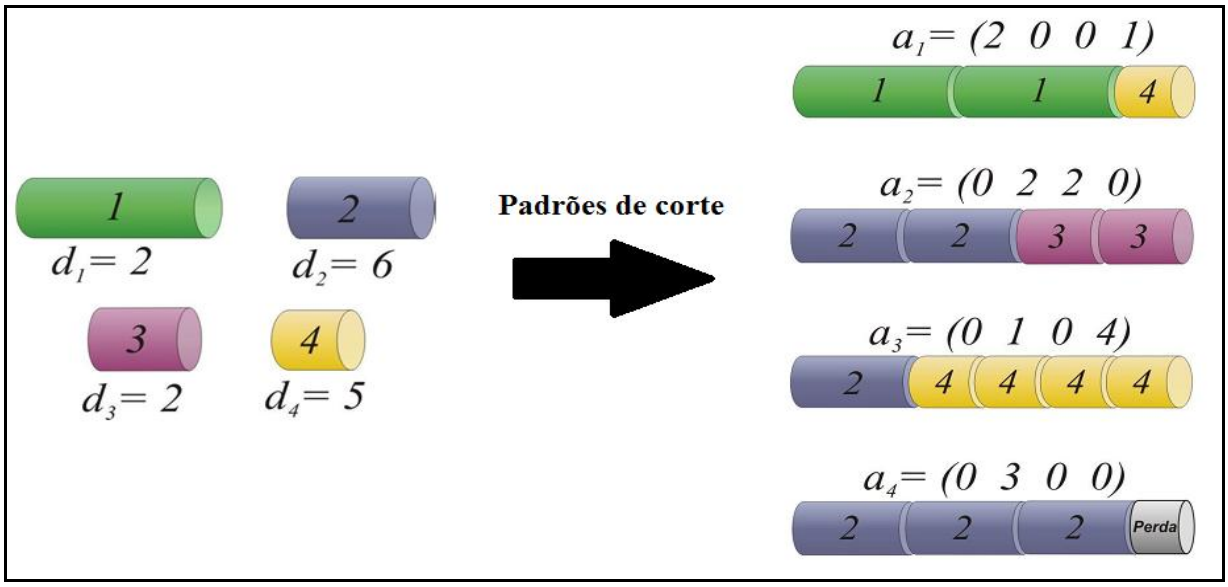

FIGURA 1 – Exemplos de possíveis padrões de corte para o PCE.

No exemplo da Figura 1, são apresentados 4 possíveis padrões de cortes para atender a demanda. O padrão de corte a_1 gera 2 itens do tipo 1 e 1 item do tipo 4. O padrão de corte a_2 gera 2 itens do tipo 2 e 2 itens do tipo 3. E o padrão de corte a_3 gera 1 item do tipo 2 e 4 itens do tipo 4. Esses três padrões de corte não geram perda. Já o padrão de corte a_4 gera 3 itens do tipo 2 e gera perda.

A solução para o problema de corte exemplificado na Figura 1 deve determinar quantos objetos serão cortados segundo cada um dos quatro padrões de corte fornecidos, de modo a atender exatamente à demanda de cada um dos itens e, ao mesmo tempo, minimizar o custo de cortar os objetos.

Para a modelagem matemática do PCE descrito, consideram-se os parâmetros e variáveis a seguir:

Parâmetros:

- S: número de tipos de objetos. Denotamos objetos tipo $s \in \{1, \ldots, S\}$;
- L_s: comprimento do objeto tipo s, $s = 1, \ldots, S$;
- e_s: número de objetos tipo s em estoque, $s = 1, \ldots, S$;
- m: número de tipos de itens demandados;
- $_i$: comprimento do item tipo i, $i = 1, \ldots, m$;
- d_i: demanda do item tipo i, $i = 1, \ldots, m$;
- J_s: conjunto de padrões de corte para o objeto tipo s, $s = 1, ..., S$;

- a_{ijs}: número de itens tipo i cortados no padrão j para o objeto s, $i = 1, ..., m$;
- c_{js}: custo de cortar o objeto s de acordo com o padrão de corte j, $j \in \{J_s, s = 1, \ldots, S\}$.

Variáveis:

x_{js}: número de objetos tipo s cortados segundo o padrão de corte j, $j \in \{J_s, s = 1, \ldots, S\}$.

Modelo Matemático:

$$\text{Minimizar } f(\mathbf{x}) = \sum_{s=1}^{S} \sum_{j \in J_s} c_{js} x_{js} \quad (1)$$

Sujeito a:

$$\sum_{s=1}^{S} \sum_{j \in J_s} a_{ijs} x_{js} = d_i, \quad i = 1, \ldots, m \quad (2)$$

$$\sum_{j \in J_s} x_{js} \leq e_s, \quad s = 1, \ldots, S \quad (3)$$

$$x_{js} \geq 0, \text{ inteiro}, j \in \{J_s, s=1,\ldots,S\} \quad (4)$$

No modelo (1)-(4) a principal decisão a ser tomada consiste em determinar a frequência com que cada padrão de corte deve ser cortado (modelo matemático orientado ao padrão). A função objetivo (1) minimiza os custos de cortar objetos segundo cada um dos padrões de cortes encontrados. As restrições (2) asseguram que a quantidade de itens cortados atenda às demandas. As restrições (3) garantem que a quantidade de objetos utilizados durante o processo de corte não seja superior à quantidade disponível em estoque. As restrições (4) referem-se à integralidade das variáveis.

3. Procedimento heurístico

Devido às condições de integralidade das variáveis (restrições (4)), é muito difícil resolver o modelo (1)-(4) na otimalidade. Neste caso, para a resolução do modelo, o seguinte procedimento foi realizado:

- Resolver o modelo com as condições de integralidade (4) relaxadas utilizando a técnica de geração de colunas (Gilmore e Gomory (1963));

- Após obter a solução ótima relaxada, com os padrões de corte gerados no passo anterior, resolver o modelo considerando as condições de integralidade (4). Note que nesta etapa não é necessário utilizar a técnica de geração de colunas.

Toda a implementação da heurística para a resolução do modelo (1)-(4) foi desenvolvida utilizando a linguagem OPL do software CPLEX. Os testes computacionais foram realizados em um microcomputador Intel Core i5 (1.60 GHz, com 6 GB de memória RAM).

4. Testes computacionais

Para avaliar o desempenho computacional da implementação do procedimento heurístico utilizado, testes computacionais foram realizados com problemas gerados aleatoriamente. Os testes realizados consideram um estoque com apenas um tipo de objeto, de dimensão 1500 cm e em quantidade suficiente para atender a demanda.

Duas classes de exemplos foram geradas aleatoriamente para o tamanho e demanda dos itens: GG (itens grandes com demanda grande) e PP (itens pequenos com demanda pequena). O intervalo no qual foram gerados os tamanhos e demandas dos itens é apresentado na Tabela 1.

TABELA 1 - Intervalo de geração de valores para as classes de exemplos.

Classe	Tamanho	Demanda
Grande (G)	[300, 700]	[50, 10]
Pequeno (P)	[140, 300]	[1, 10]

Quando se trata de problemas de programação inteira ou programação inteira mista, o ambiente de desenvolvimento do software CPLEX permite a alteração de diversas configurações relativas à resolução desses problemas, como os possíveis tipos de planos de corte e a estratégia de seleção de nós utilizada no método *branch-and-cut*. Nos testes realizados, consideramos duas estratégias de seleção de nós: busca em profundidade e busca por melhor limitante. Essas estratégias foram combinadas com três planos de corte:

- Cortes fracionários de Gomory: dado o tableau de uma base ótima de um problema relaxado, esses cortes são gerados pela aplicação de arredondamentos inteiros nas linhas relativas às variáveis inteiras que tenham valor fracionado**;**
- Cortes *Zero-half*: se baseiam na premissa de que, quando o lado esquerdo de uma desigualdade consiste em variáveis e em coeficientes inteiros, então o lado direito poderá ser arredondado para baixo;

- Cortes MIR (*Mixed Integer Rounding*): são gerados através do arredondamento inteiro dos coeficientes das variáveis inteiras e do valor no lado direito de uma restrição.

Dessa forma, foram analisadas 6 configurações:

- Configuração 1: aplicando cortes fracionários de Gomory com busca por melhor limitante;
- Configuração 2: aplicando cortes fracionários de Gomory com busca em profundidade;
- Configuração 3: aplicando cortes *Zero-half* com busca por melhor limitante;
- Configuração 4: aplicando cortes *Zero-half* com busca em profundidade.
- Configuração 5: aplicando cortes MIR com busca por melhor limitante;
- Configuração 6: aplicando cortes MIR com busca em profundidade.

Para a classe de testes GG, foram considerados 60 tipos de itens. A Tabela 2 apresenta as soluções contínuas e inteiras obtidas, o tempo computacional, o número de iterações realizadas e o GAP para cada uma das configurações testadas.

O tempo computacional mostrado nesta tabela se refere apenas ao tempo de resolução da segunda etapa do procedimento heurístico, em que as variáveis do problema são inteiras. Isso porque as alterações nas configurações do CPLEX são aplicadas apenas na resolução de problemas inteiros ou inteiros mistos.

TABELA 2 – Soluções e desempenho computacional para a classe de testes GG.

Configuração	**Solução Contínua**	**Solução Inteira**	**Tempo computacional (em segundos)**	**Iterações**	**GAP**
1	6624,24	9939	0,17	55	0,00%
2	6624,24	9939	0,17	55	0,00%
3	6624,24	9939	0,63	4288	3,16%
4	6624,24	9939	1,53	15726	2,95%
5	6624,24	9939	0,92	5737	3,17%
6	6624,24	9939	1,30	25873	3,17%

Analisando a Tabela 1, é possível perceber que, para um problema GG com 60 itens, em todas as configurações, o procedimento heurístico encontrou uma solução inteira com perda igual a 9939, o que representa um aumento de aproximadamente 50% na perda encontrada para a solução contínua.

Com relação ao desempenho computacional, o CPLEX conseguiu resolver o problema inteiro em um tempo extremamente curto. O pior caso durou apenas 1,53 segundos. Porém,

ainda é possível verificar que, para as configurações em que foi permitido gerar cortes fracionários de Gomory (configurações 1 e 2), o tempo de resolução foi ainda menor (0,17 segundos).

A maior diferença que pode ser notada na Tabela 1 é no número de iterações realizadas. Nas duas primeiras configurações, quando foi permitido gerar cortes fracionários de Gomory, o CPLEX precisou de 55 iterações para chegar na melhor solução inteira com os padrões disponíveis. Enquanto que, nas configurações 3, 4, 5 e 6, o CPLEX precisou de 4288, 15726, 5737 e 25873 iterações, respectivamente.

Na comparação entre as estratégias de seleção de nós, por melhor limitante e por profundidade, não houve influência dessas estratégias nas configurações 1 e 2. Porém, em todas as outras, a estratégia por melhor limitante se mostrou mais eficiente, realizando, no total, quatro vezes menos iterações do que a busca em profundidade.

Esse bom desempenho do CPLEX não se repetiu para a classe de testes GG com quantidades maiores de itens. Nos testes com 70, 75 e 200 itens, o programa não encontrou soluções factíveis satisfatórias, após executar por 60 a 75 minutos. Mesmo variando as configurações relativas aos planos de corte e de estratégias de seleção de nós, o comportamento do programa não apresentou uma melhora significativa. Optou-se, inclusive, em alterar outro parâmetro do CPLEX, que incentiva a geração de cortes agressivamente, ou seja, em uma frequência maior do que na configuração padrão.

A Tabela 3 mostra a melhor solução inteira encontrada para cada uma das quantidades de itens testadas e a configuração na qual a solução foi encontrada.

TABELA 3 - Melhores soluções inteiras para a classe GG com uma quantidade maior de itens.

Quantidade de Itens	Solução Contínua	Solução Inteira	GAP	Configuração
70	23,05	419	92,33%	1 e 2
75	111,66	1182	84,60%	1 e 2
200	8909	8909	56,25%	6 e Agressivo

Nos testes considerando 70 e 75 itens, novamente as soluções obtidas nas configurações que utilizam cortes fracionários de Gomory foram as melhores. Ou seja, todas as outras configurações testadas forneceram uma solução inteira pior após mais de uma hora de execução.

Porém, nos testes com 200 itens, a melhor solução encontrada ocorreu quando determinamos que o CPLEX gerasse cortes MIR de modo agressivo. Neste caso, com as configurações 1 e 2, a solução encontrada havia sido igual a 11909, com GAP de 67,29%.

Um cenário diferente foi encontrado para a classe de testes PP. Novamente considerando 60 tipos de itens, a Tabela 4 apresenta as soluções contínuas e inteiras obtidas, o tempo computacional, o número de iterações realizadas e o GAP para cada uma das configurações testadas.

TABELA 4 – Soluções e desempenho computacional para a classe de testes PP.

Configuração	Solução Contínua	Solução Inteira	Tempo computacional (em segundos)	GAP
1	0	21901	330,55	3,13%
1 e Agressivo	0	21901	123,41	7,65%
2	0	21901	550,47	38,10%
2 e Agressivo	0	21901	198,63	41,99%
3	0	21901	76,63	0,39%
3 e Agressivo	0	21901	20,38	0,76%
4	0	21901	53,67	34,89%
4 e Agressivo	0	21901	46,69	41,55%
5	0	21901	15,75	6,15%
5 e Agressivo	0	21901	4,03	7,91%
6	0	21901	15,66	35,54%
6 e Agressivo	0	21901	14,91	20,44%

Assim como havia ocorrido para a classe GG, todas as configurações testadas chegaram à mesma solução inteira para a classe PP gerada aleatoriamente. Porém, houve uma diferença significativa no tempo computacional de resolução para cada configuração.

Ao contrário do que aconteceu para a classe GG, as configurações 1 e 2 foram as piores em termos de tempo computacional, com tempo de execução do programa superior a 100 segundos e tempo máximo de 550 segundos.

O melhor desempenho ocorreu para as configurações 5 e 6 (permitindo gerar cortes MIR), principalmente para a configuração 5 gerando cortes agressivamente. Neste caso, com tempo de resolução de apenas 4 segundos foi obtida uma solução inteira com perda igual a 21901, com um GAP de 7,91%. Permitir gerar cortes agressivamente diminuiu o tempo computacional de maneira significativa apenas em algumas configurações (1, 2 e 5).

5. Conclusões

Neste trabalho, abordamos o problema de corte de estoque (PCE) unidimensional. Para resolver este problema na integralidade, aplicamos um procedimento heurístico que utiliza todos os padrões gerados durante a resolução do problema relaxado. Este procedimento foi implementado utilizando a linguagem OPL do software CPLEX. Para os testes

computacionais, realizamos diversas alterações nas configurações do software com relação aos tipos de planos de cortes realizados e à estratégia de seleção de nós durante a resolução do problema inteiro. O objetivo destes testes foi avaliar a influência dessas configurações no desempenho computacional do CPLEX.

Os resultados obtidos mostraram que, para problemas com 60 tipos de itens grandes e demanda alta, o CPLEX foi capaz de encontrar uma solução inteira com GAP muito reduzido em menos de 1 segundo. Esse desempenho eficiente ocorreu para as configurações que permitiam a aplicação de cortes fracionários de Gomory. Com as outras configurações, o tempo computacional aumentou em até 9 vezes, apesar de continuar sendo um tempo bem aceitável. Mas a maior diferença foi vista no número de iterações realizadas pelo CPLEX, que aumentou em até 285 vezes. Com relação à estratégia de seleção de nós utilizada na aplicação do método *branch-and-cut*, a seleção de nós por melhor limitante foi mais eficiente do que a seleção por profundidade.

Uma situação completamente diferente foi encontrada quando aumentamos a quantidade de itens para o problema de corte. Depois de executar por mais de uma hora, o CPLEX encontrou apenas soluções com GAP maior que 56%. Isso se deve, principalmente, ao crescimento exponencial de ramificações a serem exploradas pelo método *branch-and-cut* com o aumento no número de variáveis e à limitação do computador utilizado para os testes. É provável que, para encontrar soluções melhores, o CPLEX precisaria executar por um período de algumas horas.

Nos testes para itens e demandas pequenas, as configurações mais eficientes foram aquelas que permitiam a geração de cortes MIR (arredondamento inteiro misto). E mais uma vez a seleção de nós por limitante se mostrou mais eficiente, principalmente em relação ao valor do GAP para as configurações 2, 4 e 6 (38%, 34% e 35%, respectivamente), que utilizaram a seleção de nós por profundidade.

Com relação à continuidade desta pesquisa, pretendemos executar uma variedade maior de testes, alterando as configurações do CPLEX para mais tipos de planos de cortes. Esses testes serão executados em outro computador, mais eficiente, para garantir que o programa seja executado por longos períodos de tempo. Com isso, teremos uma visão mais ampla e completa das ferramentas e configurações disponíveis nesse solver.

Referências

CHERRI, A. C.; ARENALES, M. N.; YANASSE, H. H. The one-dimensional cutting stock problem with usable leftover – A heuristic approach. *European Journal of Operational Research*, v. 196, n. 3, p. 897-908, 2009.

CHERRI, A. C.; ARENALES, M. N.; YANASSE, H. H. The usable leftover one-dimensional cutting stock problem – a priority-in-use heuristic. *International Transactions in Operational Research*, v. 20, n. 2, p. 189-199, 2013.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting – stock problem. *Operations Research*, v. 9, n. 6, p. 849-859, 1961.

GILMORE, P. C.; GOMORY, R. E. A linear programming approach to the cutting stock problem – Part II. *Operations research*, v. 11, n. 6, p. 863-888, 1963.

HAESSLER, R. W. Controlling cutting pattern changes in one-dimensional trim problems. *Operations Research*, v. 23, p. 483-493, 1975.

HAESSLER, R. W. A note on computational modifications to the Gilmore-Gomory cutting stock algorithm. *Operations Research*, v. 28, p. 1001-1005, 1980.

HINXMAN, A. The trim-loss and assortment problems: a survey. *European Journal of Operational Research*, v. 5, n. 1, p. 8-18, 1980.

STADLER, H. A one-dimensional cutting stock problem in the aluminium industry and its solution. *European Journal of Operational Research*, v. 44, n. 2, p. 209-223, 1990.

VAHRENKAMP, R. Random search in the one-dimensional cutting stock problem. *European Journal of Operational Research*, v. 95, p. 191-200, 1996.

WÄSCHER, G.; GAU, T. Heuristics for the integer one-dimensional cutting stock problem: a computational study. *OR Specktrum*, v. 18, p. 131-144, 1996.